Dynamic XML–Based Exchange of Relational Data: Application to the Human Brain Project

Zhengming Tang[1,2] **Yana Kadiyska**[2] **Hao Li**[1]
Dan Suciu, PhD[2] **James F. Brinkley, MD, PhD**[1,2]

[1]**Structural Informatics Group**
Depts. of Biological Structure and Medical Education and Biomedical Informatics
University of Washington,
Seattle, WA

[2] **Dept. of Computer Science and Engineering**
University of Washington,
Seattle, WA

This paper discusses an approach to exporting relational data in XML format for data exchange over the web. We describe the first real-world application of SilkRoute, a middleware program that dynamically converts existing relational data to a user-defined XML DTD. The application, called XBrain, wraps SilkRoute in a Java Server Pages framework, thus permitting a web-based XQuery interface to a legacy relational database. The application is demonstrated as a query interface to the University of Washington Brain Project's Language Map Experiment Management System, which is used to manage data about language organization in the brain.

INTRODUCTION

XML has become the de-facto standard format for universal data exchange because it allows data to be exchanged regardless of the platform on which it is stored or the data model in which it is represented. However, the source data to be shared are often relational, and previous efforts at converting relational data to XML have been restrictive or inefficient. Yet the ability to exchange relational data is increasingly becoming a need, as illustrated by the Human Brain Project (HBP), which has focused on developing informatics tools designed to manage the massive amount of information that is accumulating about the brain[1]. As a result of the initial phases of this national effort, several groups have built or are building Experiment Management Systems (EMS) or other information management systems that utilize standard relational databases. As these databases mature, they must be interrelated in order to permit widespread information sharing that is essential to developing a comprehensive understanding of brain function.

Currently many of these EMS store their data in some relational database system, and offer a Web interface access to authorized remote users, returning answers as HTML pages. But this does not permit data exchange. For example, in order to process data from several EMS in a user's own application, she needs to first retrieve the data manually from each Web interface, then parse the resulting HTML pages in her application, and only then process the data. Clearly this is impractical and does not scale beyond a few EMS. In an effort to facilitate this process, we explored different methods of data exchange. One option entails the construction of a unique, centralized database incorporating data from different Human Brain Project databases and can be accessed by all users through a database query language like SQL. We soon realized, however, that scalability would become an issue as more database "nodes" are added to the picture. Another option is the complete replacement of relational databases with pure XML databases. This solution would allow users to retrieve the data automatically using an XML query language (XQuery or XPath) and would simplify the task of reading the data into the applications. However this solution is impractical for researchers who currently rely on their relational databases and have built their own proprietary systems using those databases.

We then arrived at a more promising third option: keep local data in relational databases, and add the ability to dynamically export the data in XML format using SilkRoute. SilkRoute is a middleware software layer being developed at UW[2] to export relational data to XML and thus allow free exchange of data between independent applications. This solution has several advantages: it uses the XML format for representing and exchanging data, it queries the data with XQuery, and it allows users to continue to store and manage their data in relational databases.

In this paper, we describe the first application of SilkRoute to a real-world problem: the publication of an EMS developed by the UW Human Brain Project for managing cortical language map data. We describe the existing EMS, the process of mapping the relational data to XML, and the results of several sample queries.

THE UW HBP BRAIN PROJECT EMS

The UW HBP is developing methods for organizing and managing human brain mapping data around a structural (anatomical) information framework, with application to understanding language organization in the cortex. The primary data are obtained at the time of neurosurgery for intractable epilepsy and from functional imaging studies performed prior to surgery. A variety of other data, stored in other databases, represent structural information, images, and single-cell recordings. A goal of the informatics application is to integrate these diverse forms of data. The EMS is a web-based tool for organizing and managing these data. It is built using a toolkit called WIRM[3] and uses Perl scripts to publish, as HTML pages, relational data stored in a MySQL database. However, prior to this project the EMS did not include a general-purpose query language that would make its data available to other applications. It was therefore a good candidate for exploring the utility of SilkRoute as a means for exporting and querying the EMS data in XML.

XBRAIN: USING SILKROUTE TO PUBLISH THE HBP EMS DATA IN XML

As mentioned, the purpose of SilkRoute is to publish relational data as XML. This approach is beneficial for many reasons. First, most data are already stored in relational databases; SilkRoute uses that data “as is”. Second, all queries are evaluated by SilkRoute entirely within the relational engine, which leads to high performance because it benefits from the wide range of techniques available today in relational engines: indexes, cost-based query optimizers, data statistics, buffer managers, transaction processors. Last but not least, SilkRoute provides a dynamic, structured view of the data.

Querying in SilkRoute is easy. SilkRoute treats the relational database as an XML document with the DTD shown in Figure 1.

```
<! ELEMENT Database[
    <!ELEMENT Relation1(attr1,attr2….)>
    <!ELEMENT Relation2(attr1,attr2….)>
    <!ELEMENT Relation3(attr1,attr2….)>
]>
```

Figure 1. Relational Database

The database administrator can restructure this document or hide part of the information from the public by defining a view over the database. He/she does this by creating a public view over the relational database (this view is very similar to the concept of relational view, only it is written in XQuery instead of SQL). This public view, much like relational views, is not materialized. Instead, its Document Type Definition (DTD) is presented to the users, who write their XQueries over the public view’s DTD.

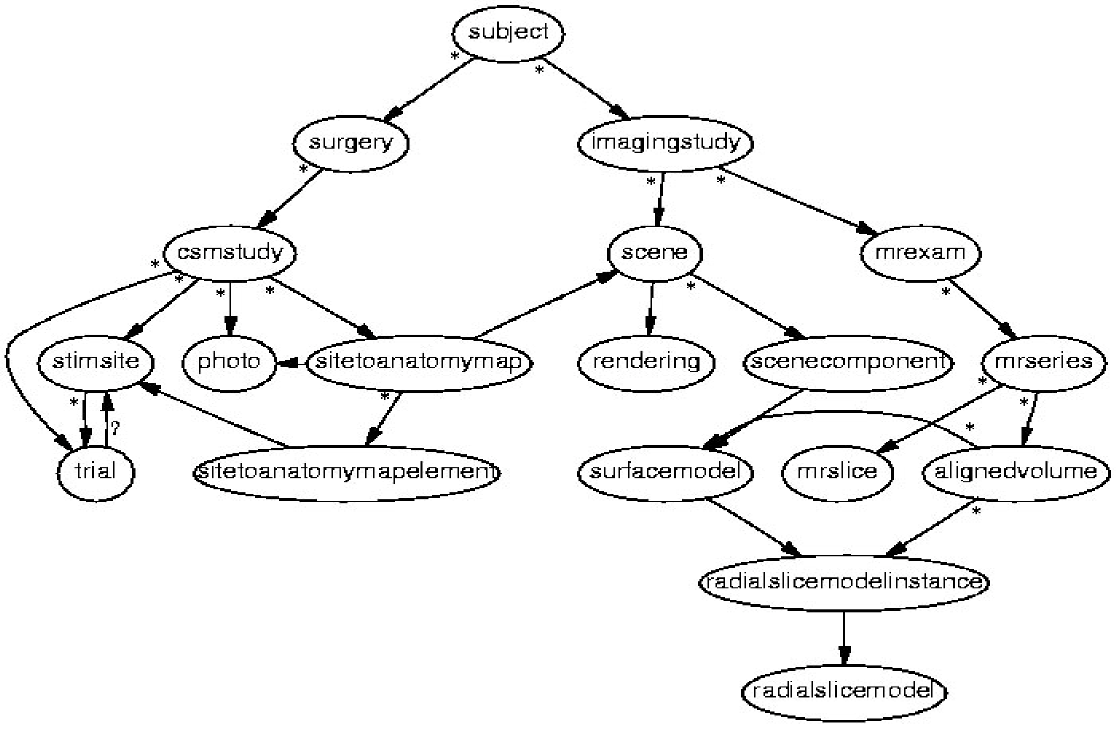


Figure 2. DTD graph of Public View (showing major elements

```
<root>
{   for $subject in DATABASE/Patient
    return
      <subject oid="{$subject/oid/text()}">
         <initials> {$subject/initials/text()} </initials>
         <first_name> {$subject/first_name/text()} </first_name>
           ...
           {for $surgery in DATABASE/Surgery
            where data($surgery/patient) = data($subject/oid)
            return
               <surgery oid="{$surgery/oid/text()}">
                  <surgery_date> {$surgery/surgery_date/text()} </surgery_date>
                  <surgeon> {$surgery/surgeon/text()} </surgeon>
                  ...
```

Figure 3. Fragment of Public View

SilkRoute then composes the user query with the public view and evaluates the final answer.

For the XBrain application, consider the following subset of relations present in our database: *Patient, Surgery, CSMStudy…* The database administrator decides to expose these tables as an XML public view with the structure shown Figure 2. Note for this public view, she decides to rename the top element to *subject* instead of using the database name *patient*.

The administrator then specifies this public view as a large XQuery query, a fragment of which is shown in Figure 3 (DATABASE refers to the relational database—see Figure 1). What is important is that the administrator has the freedom to define and structure the XML Public View totally differently from the relational schema: all she needs to do is to define some XQuery query that transforms the relational data into the XML Public View. For example, she changed the name from *patient* to *subject*. Presently, creating this public view is done manually. We are currently working on automating this process.

The last step in the development of XBrain was to make the application available over the web. We decided to write the web application in Java/JSP and use Apache Tomcat[4] as the application server. The Java code makes calls to SilkRoute through a Java interface and gives SilkRoute all the necessary inputs (relational schema, public view, user query). SilkRoute then queries the database, leaving all data processing to the relational engine, and returns a string containing the XML output (see Figure 4). This string is displayed to the user directly in their browser, and some browsers (i.e. Internet Explorer) will present this XML document in a clear, hierarchical format.

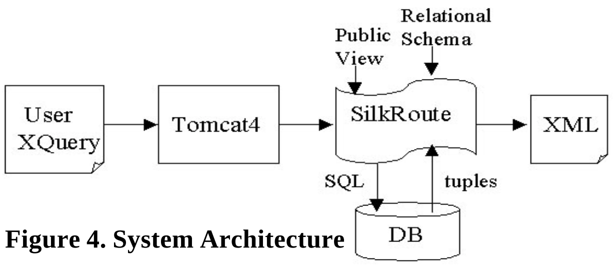


Figure 4. System Architecture

Furthermore, the database administrator implemented an additional public view that hides patient sensitive data from public users. Queries from users without special permissions are run over this public view. Only when a user with the correct permissions is authenticated by the web application (with username and password) are their queries run over the public view that exposes the entire patient data set.

XBRAIN: SAMPLE QUERIES AND RESULTS

One could evaluate the XQuery defining the Public XML View: this is possible in SilkRoute, but is seldom done, because it results in a large XML document representing the entire relational database. Instead, the Public View is kept *virtual*, and users access it by formulating XQuery queries over this view, which typically return only small XML fragments. To illustrate our application, the following sample queries were run in XBrain. We start with a relatively simple example.

Example 1:
List last names of all subjects whose age is greater than 20.

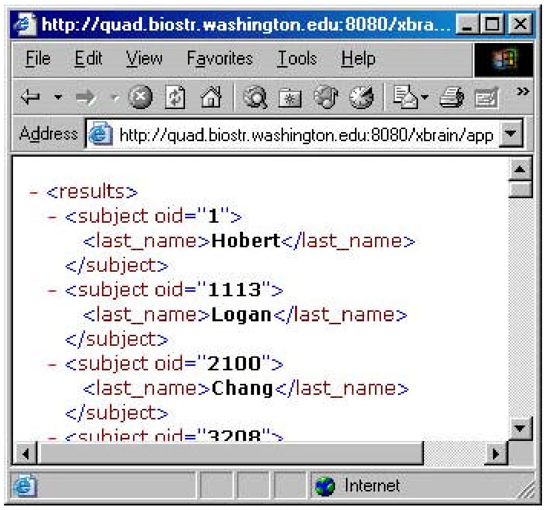
User Query (written in XQuery, note $pv refers to the Public View XQuery, shown in Figure 3.):

```
<results>
{   for $s in $pv/root/subject[age>20]
    return
       <subject oid="{$s/oid/text()}">
          <last_name>
            {$s/last_name/text()}
          </last_name>
       </subject>
}</results>
```

SilkRoute composes this query internally with the Public View XQuery, and generates the following SQL query, which is not seen by user, but instead executed on the database:

```
SELECT P89.oid, P89.last_name
FROM Patient as P89
WHERE P89.age > 20;
```

The XML result is shown in Figure 5 (note no real names are stored in the database, only pseudonyms).


http://quad.biostr.washington.edu:8080/xbra...
File Edit View Favorites Tools Help
Address http://quad.biostr.washington.edu:8080/xbrain/app
Internet

```
- <results>
  - <subject oid="1">
      <last_name>Hobert</last_name>
    </subject>
  - <subject oid="1113">
      <last_name>Logan</last_name>
    </subject>
  - <subject oid="2100">
      <last_name>Chang</last_name>
    </subject>
  - <subject oid="3208">
```

Figure 5. Example 1 XML Results

The next example illustrates a more complex query.

Example 2:
List identifiers of all patients that had one language error of type "semantic paraphasia" (semantic paraphasia is coded as “2” in the miriam_code column of a Trial). For each of these patients, list the specific stimulation sites and corresponding anatomical locations where these errors occurred.

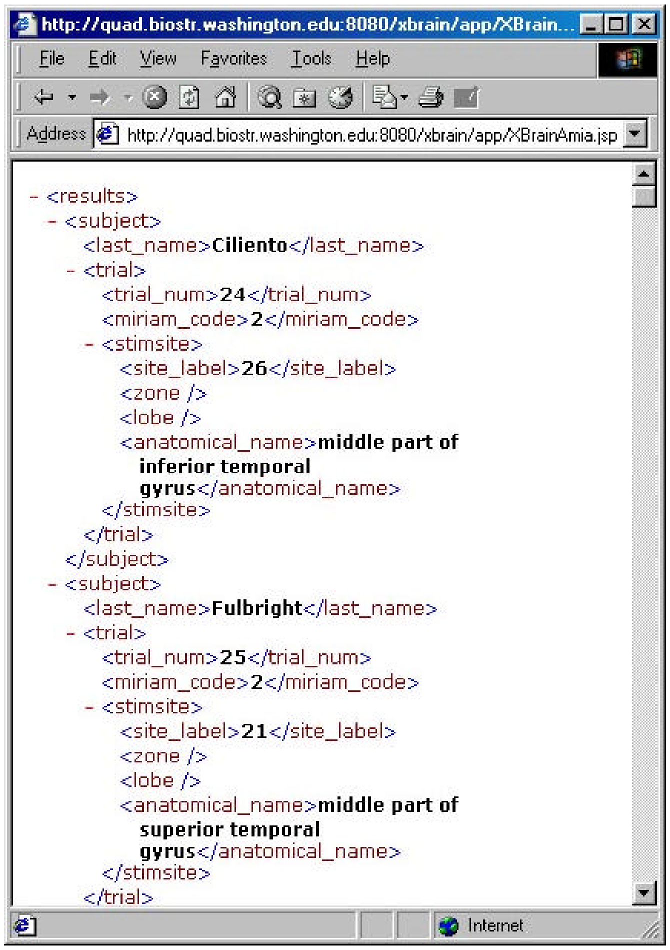
The user XQuery for this example is shown in Figure 6a., and the intermediate SQL produced by SilkRoute is shown below:

```
SELECT P89.last_name, T259.miriam_code,
        T259.trial_num, S269.lobe,
        S269.site_label, S269.zone,
        S269.anatomical_name, S269.oid,
        C254.oid,S244.oid,T259.oid,P89.oid
FROM CSMStudy as C254, Surgery as S244,
       Trial as T259, StimSite as S269,
       Patient as P89
WHERE S269.oid = T259.stimulation_site AND
       T259.csmstudy = C254.oid AND
       S244.patient = P89.oid AND
       C254.surgery = S244.oid AND
       T259.miriam_code = ‘2’ AND
       EXISTS (
          SELECT T222.miriam_code
          FROM CSMStudy as C213,
                Surgery as S203,
                Trial as T222
          WHERE S203.patient = P89.oid AND
              C213.surgery = S203.oid  AND
              T222.csmstudy = C213.oid AND
              T222.miriam_code = ‘2’ );
```

The XML result returned is shown in Figure 6b.

```
<results>
{  for $p in $pv/root/subject
   where $p/surgery/csmstudy/trial/miriam_code/text()="2"
   return
      <subject>
         <last_name>{$p/last_name/text()}</last_name>
         {  for $t in $p/surgery/csmstudy/trial
            where $t/miriam_code/text()="2"
            return
               <trial>
                  <trial_num>{$t/trial_num/text()}</trial_num>
                  <miriam_code>{$t/miriam_code/text()}</miriam_code>
                 {  for $s in $t/t_stimsite
                    return
                       <stimsite>
                          <site_label>{$s/t_site_label/text()}</site_label>
                          <zone> {$s/t_zone/text()}</zone>
                          <lobe> {$s/t_lobe/text()}</lobe>
                         <anatomical_name>{$s/t_anatomical_name/text()}</anatomical_name>
                       </stimsite>
                 }
               </trial>
         }
      </subject>
}</results>
```

Figure 6a. Example 2 User Query


http://quad.biostr.washington.edu:8080/xbrain/app/XBrain...
File Edit View Favorites Tools Help
Address http://quad.biostr.washington.edu:8080/xbrain/app/XBrainAmia.jsp
Internet

```
- <results>
  - <subject>
      <last_name>Ciliento</last_name>
    - <trial>
        <trial_num>24</trial_num>
        <miriam_code>2</miriam_code>
      - <stimsite>
          <site_label>26</site_label>
          <zone />
          <lobe />
          <anatomical_name>middle part of
            inferior temporal
            gyrus</anatomical_name>
        </stimsite>
      </trial>
    </subject>
  - <subject>
      <last_name>Fulbright</last_name>
    - <trial>
        <trial_num>25</trial_num>
        <miriam_code>2</miriam_code>
      - <stimsite>
          <site_label>21</site_label>
          <zone />
          <lobe />
          <anatomical_name>middle part of
            superior temporal
            gyrus</anatomical_name>
        </stimsite>
      </trial>
```

Figure 6b. Example 2 XML Results

DISCUSSION

In this paper we discuss an XML based approach to exchanging relational data from the Human Brain Project or other relational databases. Specifically, we applied this technique to the UW Brain Project EMS and used a web interface to send queries and retrieve XML results from SilkRoute.

SilkRoute offers several major benefits. First, it can dynamically and efficiently generate the XML data. Second, this process can be applied to any database and its data can be mapped to any number of DTD/XML schemas. With published XML schemas that can serve as an interface for mediating query and data exchange (for example, BDML from Dan Gardner, Cornell Univ.[5]), this approach should facilitate data exchange between researchers without hindering their own local, independent efforts.

One shortcoming of XBrain is that the current user interface is targeted towards researchers with some programming background. A user is required to formulate XQueries and read XML results. To expand the user base to all potential researchers, we are exploring the implementation of a novel graphical user interface to formulate queries, and flexible formats for visualizing results using XSLT.

Other future work will extend these same techniques to other databases in the Human Brain Project and allow multiple applications to cooperate in a peer data management system. We are currently investigating Piazza[6], a framework that will use SilkRoute as the infrastructure for peer-to-peer data management.

ACKNOWLEDGEMENTS

The research for this publication was funded by Human Brain Project grant RO1 MH/DC02310, National Institute of Mental Health, and the National Institute of Deafness and Other Communication Disorders.

REFERENCES

1. Human Brain Project, http://www.nimh.nih.gov/neuroinformatics/index.cfm

2. Fernandez M, Kadiyska Y, Morishima A, Suciu D, Tan W. SilkRoute: A Framework for Publishing Relational Data in XML. ACM Transactions on Database Technology, vol. 27, no. 4, December, 2002

3. Jakobovits, R. M., C. Rosse, et al. (2002). "An open source toolkit for building biomedical web applications." J Am Med Ass. 9(6): 557-590.

4. http://jakarta.apache.org/tomcat/

5. Gardner D, Knuth KH, Abato M, Erde SM, White T, DeBellis R, Gardner EP. Common data model for neuroscience data and data model exchange. J Am Med Ass 2001;8(1):17-33.

6. Gribble S, Halevy A, Ives Z, Rodrig M, Suciu D. What can databases do for peer-to-peer? In: WebDB Workshop on Databases and the Web; 2001. http://data.cs.washington.edu/papers/p2p.pdf.